Final Project Design

Team 21

Austin Clum, Rachel Elting, Gwen Liu, Nick North, Sarah Scott

Project Name

Cosplay Companion

Project Synopsis

The Cosplay Companion is an electronic costume helmet paired with an integrated companion app, allowing for unique LED expression puppeteering and heat-monitoring safety features.

Project Description

A new development in cosplay is the use of electronics such as LEDs and servos to enhance various features of a costume.

A frequent issue of heavier costumes is the danger of heat exhaustion. This is particularly concerning when a costume involves a helmet, because it is important to dissipate heat away from the head. The use of electronics in a costume provides the unique opportunity to add sensors into a helmet that can detect excessive heat levels, and with the prevalence of IoT-compatible microcontrollers, it is possible to sync the sensors and other electronic components with a custom mobile app.

The Cosplay Companion aims to test this concept by constructing an electronic costume helmet with an integrated companion app. The helmet will display expressions and animations on LED panels that can be controlled manually by the wearer via buttons in the gloves, allowing for exceptional puppeteering and audience interaction. The app will use heat/humidity readings from the sensors to control fans and alert for excessive heat levels. In addition to these safety features, the app may provide useful control options for the LEDs, such as color adjustment, extra animation features, and switches for battery-saving or photo-friendly modes.

The project will result in a fully functional costume helmet that can be worn to fan conventions and maker events. It also serves as a prototype if any group members would like to take commissions in the future.

Project Milestones

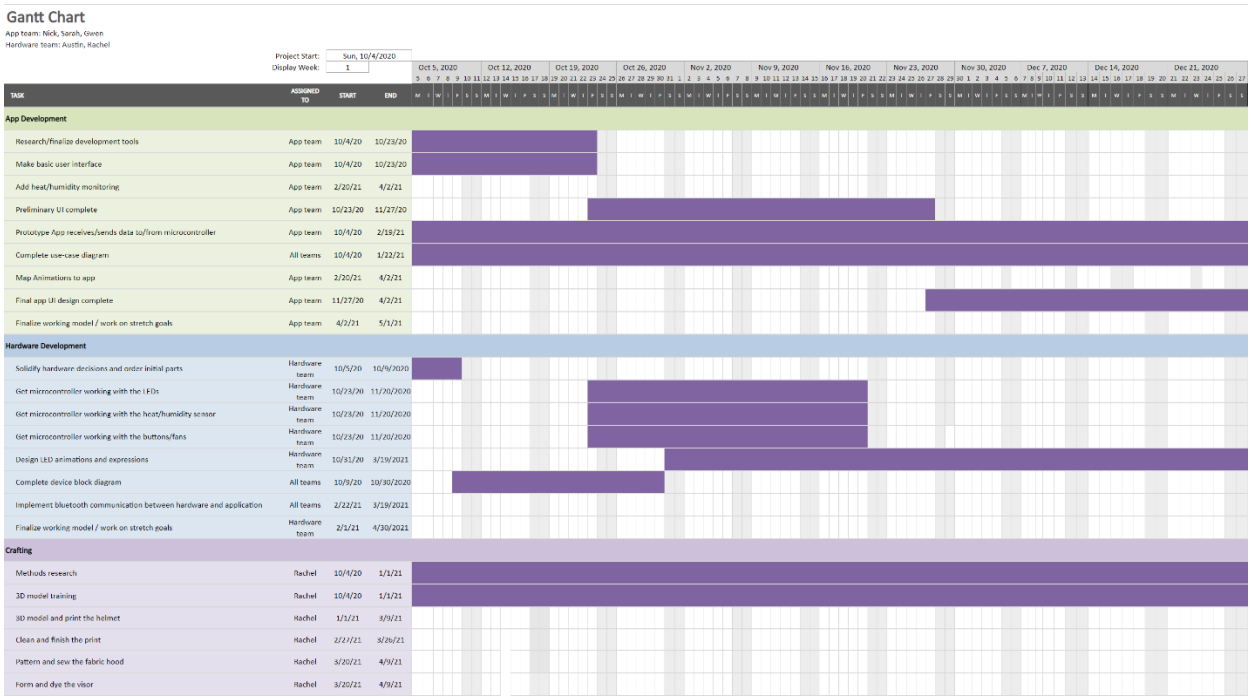
Milestones:

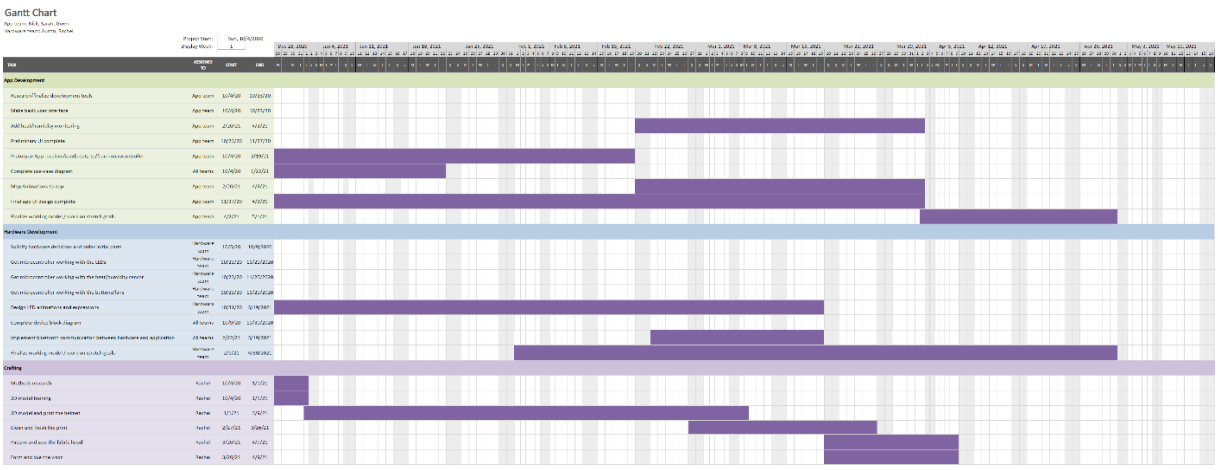
- Fall semester
    - All hardware/software decisions finalized (10/23/20)
    - Device block diagram complete (10/30/20)
    - Microcontroller sends/receives data to/from peripherals (11/20/20)
    - Preliminary UI for app complete (11/27/20)
    - Use case diagram complete (12/04/20)
- Spring semester
    - Prototype: App receives/sends data to/from microcontroller and peripherals (2/19/21)
    - All required hardware implemented (3/12/21)
    - Physical helmet base complete (3/26/21)
    - Final app UI design complete (4/2/21)
    - Final testing complete (5/5/21)

Gantt chart:

Our work plan is to have an app team consisting of Nick, Sarah, and Gwen, as well as a hardware team consisting of Austin and Rachel, while Rachel takes primary responsibility for the crafting. The following Gantt chart shows the predicted timeline of tasks to complete. Each task is separated into the project's three main categories: the app, the hardware, and the crafting.

Please refer to the attached PDF for a full-size view of the Gantt chart.





Project Budget

Hardware, software, and/or computing resources:

This project is split into three main parts: the app, the hardware, and the crafting. Hardware and software decisions are made with an emphasis on the availability of learning resources within the constraint of being budget friendly.

The app will make use of a cross platform JavaScript framework so that we can develop for both iOS and Android devices.

Software for the helmet will be developed using the Arduino IDE and open-source APIs such as the hzeller library.

For 3D modeling, we will use the personal use license for Fusion 360.

Estimated cost:

Below is an updated budget estimate, pending approval:

Hardware:

- Raspberry Pi 3 B+: $35.00
- RGB Matrix Bonnet for Raspberry Pi: $14.95
- 64x32 RGB matrix 3mm pitch: $44.95; x2 = $89.90
- 16x8 1.2” LED matrix + backpack: $21.95; x2 = $43.90 (stretch goal)
- Temperature/humidity sensor: $8.95
- Portable 5V 4.8A power supply: $49.99
- 5v Fans x2: $6.99
- Tactile switch buttons x10: $2.50
- 1.14” LCD display: $9.95 (stretch goal)

- Hall sensor: $2.00 (stretch goal)
- ESP32 development board: $9.99; x2 = $19.98
- 10k resistors pack of 25: $0.75
- LiPo battery: $14.95; x2 = $29.90
- LiPo charger: $6.95

Software:

- No anticipated software costs.

Crafting:

- Fabric 1yd: $25.99; x2 = $51.98
- EVA foam roll: $9.99
- Vacuum forming:
    - PETG sheets: $32.74
    - Forming services: ???
    - iDye Poly: $4.49; x3 = $13.47
- Finishing materials:
    - Safety equipment
        - Respirator: $31.47
        - Gloves: $2.48
    - Materials
        - Primer: $6.99
        - Base coat: $5.98
        - Bondo: $6.27
        - Sandpaper, various sizes: ~$25.00
        - Clearcoat: $12.78
        - Chrome: $12.99; x2 = $25.98
        - Airbrush kit: $38.99

Cost estimate for ideal parts list:
Total for first helmet: $625.83
Total for second helmet (assuming sharable parts): $509.13
Total for both: $1,134.96 + vacuum-forming services

Note that the above is an ideal parts list that may be adjusted pending budget approval. Off-brand alternatives at lower costs may be considered, stretch goal peripherals may not be used, and simpler crafting alternatives may be opted for.

Due to concerns regarding COVID-19, we plan to make two helmets so that more team members may work on hardware while maintaining physical distance.

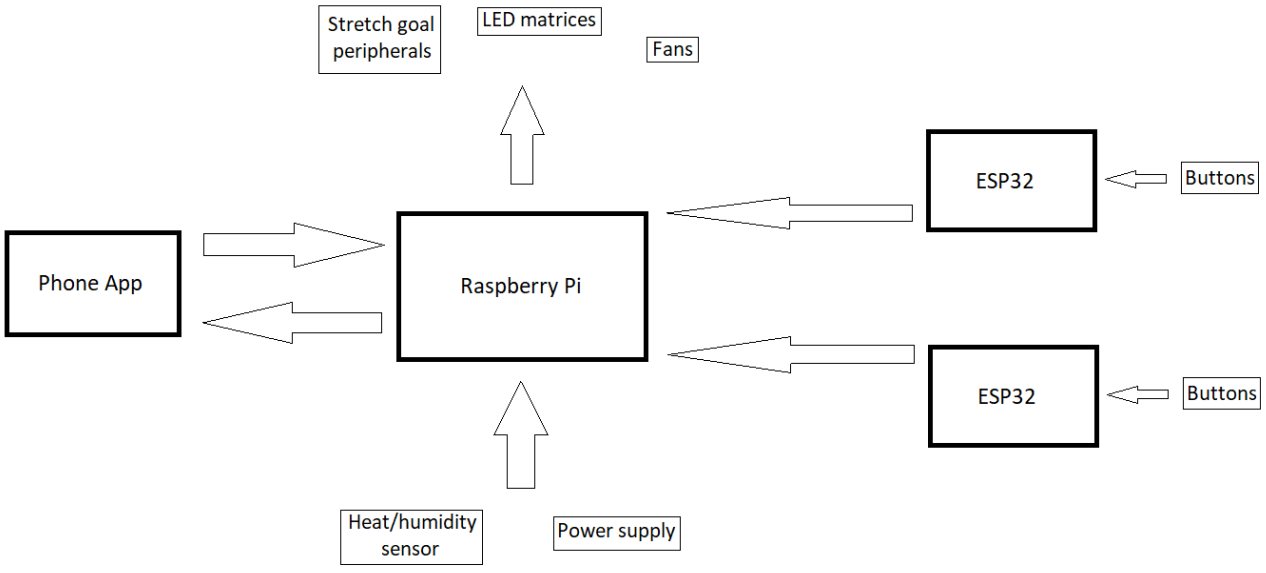
Vendors:
None of the preferred vendors carry any required parts, so they must be acquired from other sources. The above prices are from vendors Adafruit, MicroCenter, Amazon, BigZFabric, and Home Depot.

Special training:
Many aspects of the project are new to the members and will require hands-on learning. We kept this in mind as we selected our hardware, software, and crafting methods so that free learning resources are available. There are no anticipated training costs.

When they will be required:
We are prioritizing the software development and hardware aspects of the project. As such, we request that parts listed under "Electronics" be supplied by 11/01/20 so that we may begin prototyping. The crafting section of the project may begin after we are satisfied with our prototype.

Final Project Design

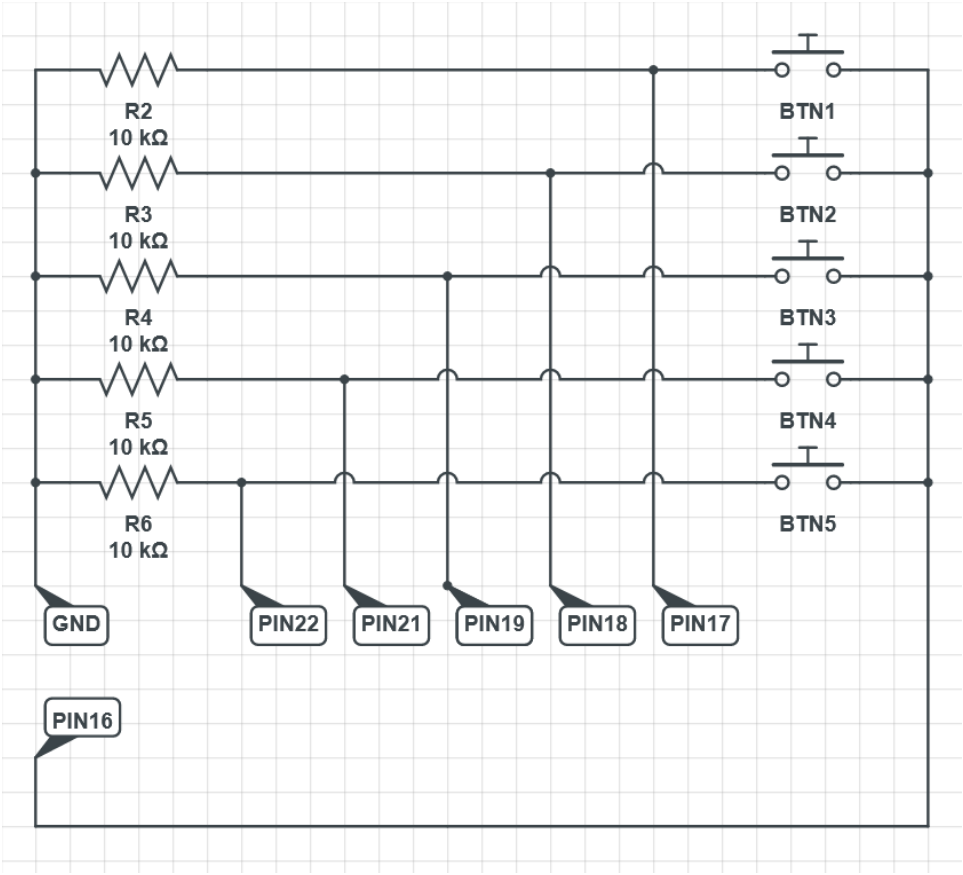
This project consists of four different devices communicating with one another: The Raspberry Pi 3B+, which drives the primary peripherals; two ESP32s, which allow for wireless communication with the buttons; and the phone app, which allows the user to view status updates and make changes to the peripherals. Please refer to Figure 1 for a simple device-block diagram that illustrates the main components of the project.



(Fig. 1): Device-block diagram of the project, showing primary devices and peripherals.

The first two devices are ESP32 microcontrollers, which will be programmed using the Arduino IDE. These chips are programmed using a subset of C and C++ which has been simplified to be both beginner-friendly and easier to fit on a small chip. This software will simply be used to detect when one of up to five buttons is pressed and send that data to the other hardware.

The devices will use the schematic shown in Figure 2 when detecting whether each button is pressed or released. Pin 16 is set to have a constant output of 5V. The other five pins are all initially connected to the ground with a 10kΩ resistor between them. This ensures that, if the pin's assigned button is not pressed, the voltage is 0V. When one of the buttons is pressed, the voltage at the pin is 5V. The software on the pin is set to run a simple loop. Each time the loop is run, the voltage at each of the five input pins is read. If the voltage is different from the previous pass through the loop, it will send the updated button status to the Raspberry Pi 3b+ through Bluetooth.



(Fig. 2): Five-button schematic for an ESP32 board.

The next device used is a Raspberry Pi 3B+. This device runs on a Linux-based operating system. It will run Python scripts to receive data via Bluetooth from the ESP32 chips and write information into a new file. In addition to the Python scripts, the Raspberry Pi will also run a C++ program that will read data from the file as well as monitor the heat and humidity sensors. The status of these sensors will be used to activate or deactivate cooling fans in the helmet according to the user's settings. The Raspberry Pi will also be responsible for running the LED matrices. For displaying images and animations, we will be using the hzeller library, which is natively programmed in C++ with options for C and Python. These matrices do not have PWM capabilities, so they require a high-speed processor to continually refresh the image. We must keep in mind that the matrices will be using a significant percentage of the Pi's CPU.

The final device is a smart phone application that will run on Android devices and ideally Apple mobile devices as well. The app will communicate with the microcontroller in order to help control the electronics. The main purpose of the app will be to allow the user to have a convenient way to change their temperature preferences and receive warnings about dangerous heat and humidity levels. The temperature and humidity levels inside the helmet can be monitored through the app. If the user wants the fans to turn on at a lower temperature, then they can easily lower the threshold temperature. The user will also be able to make changes to the animation mapping of the buttons though the app. For example, the user may frequently use the "Excited" animation, so it may be more convenient for them to only have to click one of the buttons rather than two at a time. They should be able to move around the button mapping of the animations in an easy-to-use interface. In addition, there will be a color wheel which allows the user to select the default color displayed on the helmet, as well as buttons for photo-friendly or power-saving modes. A stretch goal for both the hardware and software team to work on would be adding a larger list of animations so the user could pick some subset of these animations to map to the buttons.

There are several constraints on the software side of our project. The main tool we will be using to develop our app is Ionic. Ionic is a framework for cross platform app development with integrated libraries from Angular, React, and Vue. We chose to go with React for this project. Since we are planning to develop for both iOS and Android devices, Ionic and React are part of our development constraints. These frameworks both use JavaScript, so our language constraint for the app is that we must use JavaScript. Developing the app for both platforms makes debugging through a development environment difficult for some functionalities, so we are constrained to debugging on our mobile devices. Our only nontechnical constraint is that we need to stick to a schedule so that we can meet various project deadlines.

An important aspect of this project is to allow for exceptional puppeteering. Although animations and facial expressions can be controlled through the app, this does not allow for the quick responses required for audience interaction. The ESP32s are in place to rectify this issue – they receive input signals from buttons that are hidden in the gloves, which the user can press to quickly change expressions. Although each ESP32 can read up to 22 buttons, this may not be practical from an end user's standpoint. The button mapping will be finalized when we have a working glove prototype and can fine-tune its use. To allow the user to hold one face for a significant amount of time without requiring them to hold the button combination the entire time, the animation drawn only updates when a button is pressed, rather than released.

	No modifier	Mod1	Mod2	Mod1+Mod2
Happy	Neutral	Cheerful	Starry-eyed	Excited
Shocked	Shocked(happy)	Shocked(sad)	Teary	Bluescreen
Angry	Unimpressed	Angry	Dead	Table flip
Meme	Party parrot	Rick roll	The Game	This is fine

(Fig. 3): Potential six-button use-case chart.

Figure 3 shows a mock-up of a six-button use-case chart. This version involves four buttons in one hand that relate to a theme, as well as two modifier buttons in the other hand that can be used to access various animations within the theme. A third modifier button will be implemented which clears all modifiers since the animation does not update on button release.

One difficult constraint we face is how we will power the system using a portable battery pack. This is especially challenging for the LED matrices. The matrices are designed so that not all LEDs are lit at once, rather, it takes advantage of our persistence of vision and refreshes them quickly one row at a time. This way, we can still see the full image while the matrix minimizes its power consumption. Even so, when using an image that lights the entire display, the current draw is significant. For two matrices that are 64 pixels wide, the panels by themselves can theoretically draw 15.36 amps of current when on full bright white. This number does not include the power consumption of the Pi and other peripherals. While it is possible to find power supplies with this kind of output, they must be plugged into a power outlet, which is not feasible for a costume helmet. Portable 5V battery packs are common, such as those used to charge phones, but the current output per USB port is generally capped at 2.4A, well below the 15.36A we could theoretically use. We currently power the helmet using a battery pack that has two USB outputs -- using one to power the Pi directly, and the other to power the matrices -- under the constraint that we will use images that do not draw much power. If this does not suit our needs, we may investigate alternative options such as the use of buck converters to down regulate a battery pack with a higher voltage rating.

The final element of the project is to craft the physical helmet. The current plan is to 3D model and print the parts we need. No team members are experienced in 3D modeling, but there are free resources to learn. If we do not have time to learn, we could optionally commission another cosplayer for their .stl files for us to print. Due to concerns regarding COVID-19, we may not have access to a 3D printer when we need it. If this happens, we will craft the helmet using EVA foam. Another concern is we may not have access to a vacuum former for the visor; if we do not have the option to use local vacuum formers, we may be able to craft our own. Optionally, we can construct a makeshift visor with fabric and wire.

Ethical Issues

Humidity may cause damage:

Although there are no tutorials for waterproofing cosplay electronics, it is common knowledge that humidity may damage electrical components. Since a helmet can get humid quickly and our helmet contains electronic parts, this of great concern to our project. We would not want to sell a product that could break from normal use. The heat/humidity sensor is not only for the safety of the wearer, but for the electronics as well. It may be beneficial to have one fan near the electronics running at all times, with supplemental fans as needed. The primary source of humidity in a helmet is the wearer's breath. We can design the helmet to have a chute blocking off the wearer's nose and mouth from the electronics. With these mitigations, we hope to prevent possible damage.

Intellectual Property Issues

The cosplay itself:

Cosplay as a hobby can invoke a discussion on intellectual property, as cosplayers usually make costumes based on licensed characters. Creating a personal costume of a licensed character is generally considered a derivative work or fair use, though selling the costume may cause issues. In our case, our LED helmet is of a species from Zenith's Outer Reach, the creator of which welcomes fans to make their own characters as long as the design follows canon specifications. She even allows fans to sell original character designs and costumes. If members of our group decide to take commissions or sell helmets in the future, we should have no issues, assuming our character design is sound.

Playing games and memes:

The costume itself is not the only intellectual property we are using – we plan to use various memes that we did not originally create. One notable example is the Rick Roll. This video is often used online as a prank, and its use in an electronic costume could allow for some excellent audience interactions. However, this is a licensed music video that we did not gain explicit permission to use. The same is true for other memes we would like to add. In addition, one of our stretch goals is to hook up a gaming controller to play games such as Tetris and DOOM on the helmet. For a personal-use project, using these IPs is akin to playing games or music videos on our own personal device and constitutes as fair use. However, if we sell helmets in the future, we likely cannot include licensed content.

Change Log

Project Milestones:

Updated milestones and Gantt chart to reflect plans for current semester.

Project Budget:

Added LiPo batteries and chargers for the ESP32s. Clarified estimates for the crafting portion.

Final Project Design:

Removed implementing different animations based on length of button presses.

Animations only update on button release, allowing the user to show one animation for an indefinite amount of time without requiring them to hold a specific button combination the entire time.

Updated the plans for crafting.

Updated project constraints.